Answer Set Semantics vs. Information Term Semantics

Camillo Fiorentini, Mario Ornaghi

Dipartimento di Scienze dell'Informazione, Università degli Studi di Milano, Italy
{fiorenti, ornaghi}@dsi.unimi.it *

Abstract We compare Answer Set semantics for propositional theories and F_{cl}, a constructive logic introduced in [8].

1 Introduction

In this paper we present a preliminary comparison of the answer set semantics of programs with nested expressions (NE) [7] and a variant of the semantics of the constructive intermediate logic F_{cl}, which was introduced in [8]. This research originates from our investigation of "snapshot generation" for the OO modeling language CooML [10], a language combining the OO class based approach and a data model based on the constructive semantics of F_{cl}.

Snapshot generation has been proposed as a way of *validating* specifications in the context of the UML modeling language [3], where a snapshot is an object diagram representing (part of) the current state of a system modeled by a UML class diagram. As well-known, validation is different from *verification*. The latter proves/disproves properties of formal specifications/programs for good. The former has the purpose of showing that a formal specification meets its original informal requirements. Validation can be performed only informally through small, human readable examples or counterexamples. In the CooML language, a system S is specified by a set of formulas Cl, representing the class section of S, and a set K of constraints. The formulas of Cl are (syntactically) restricted first order formulas, while K may be any first order theory. The class section formalizes the possible information contents through the notion of "piece of information". This is a pair $\tau : Cl$, where τ is a set of data structured according to Cl, which we call an "information term" of type Cl. In CooML, an information term τ represents the data contained in a system state (i.e., a system *snapshot*), while $\tau : Cl$ represents their meaning in the problem domain. More precisely, $\tau : Cl$ can be characterized by a set of "answers" that can be true or false (in a classical interpretation). In this context, generating a snapshot means generating an information term τ such that the answers of $\tau : F$ are consistent with the constraints K.

Building snapshot generation algorithms for CooML, we have to consider computable constraint theories. In particular, we are currently investigating the possibility of representing CooML snapshot generation in the DLV system [5]. In this investigation, we noticed a similarity of the syntax of the class section of CooML systems, in the propositional fragment, with the language of nested expressions. In this paper, we present

* Work partly supported by the MIUR Project "Potenziamento e Applicazioni della Programmazione Logica Disgiuntiva".

some preliminary results, linking the answer set semantics of nested expressions with a semantics based on pieces of information.

The paper is organized as follows: in Section 2 we briefly recall nested expressions, while in Section 3 we recall the logic of pieces of information. The comparison is carried out in Section 4. Our study is still in an initial stage, and future work is briefly discussed in the conclusion.

2 Programs with Nested Expressions

We briefly recall programs with nested expressions [7]. The definitions are the usual ones. The main purpose of the section is to introduce the notation adopted in the paper. A *nested expression* is built from literals and the symbols $\top$ (true), $\bot$ (false), $\wedge$, $\vee$, *not*. [1] We say that a nested expression is *positive* iff it does not contain *not*. A *rule* with nested expression has the form $H \texttt{:-} B$, where H (the head) and B (the body) are nested expressions. A program with nested expressions (PNE) is a set Π of rules. Let X be a consistent set of literals and Π be a PNE. The satisfaction relation $X \models \Pi$ and the reduct Π^X are defined in Fig. 1.

$X \models l$	iff $l \in X$, with l a literal	l^X	$= l$
$X \models \top$		$\top^X$	$= \top$
$X \not\models \bot$		$\bot^X$	$= \bot$
$X \models A \wedge B$	iff $X \models A$ and $X \models B$	$(A \wedge B)^X$	$= A^X \wedge B^X$
$X \models A \vee B$	iff $X \models A$ or $X \models B$	$(A \vee B)^X$	$= A^X \vee B^X$
$X \models not\ A$	iff $X \not\models A$	$(not\ A)^X$	$=$ (if $X \models A$ then $\bot$ else $\top$)
$X \models A \texttt{:-} B$	iff $X \not\models B$ or $X \models A$	$(A \texttt{:-} B)^X$	$= A^X \texttt{:-} B^X$
$X \models \Pi$	iff $X \models R$, for every $R \in \Pi$	Π^X	$= \{ R^X \mid R \in \Pi \}$

Figure 1. The $\models$ relation and the reduct F^X

Answer sets, weak and strong equivalence are defined as follows:

Definition 1 (Answer Sets). *A set X of literals is an answer set of a positive PNE Π iff $X \models \Pi$ and, for every $X' \subseteq X$, $X' \models \Pi$ entails $X' = X$.*
A set X of literals is an answer set of a PNE Π iff it is an answer set of Π^X.

We denote by $\texttt{as}(\Pi)$ the set of answer sets of Π.

Definition 2 (Weak and Strong Equivalence). *Let Π_1 and Π_2 be two PNE. They are (weakly) equivalent, written $\Pi_1 \sim_w \Pi_2$, iff $\texttt{as}(\Pi_1) = \texttt{as}(\Pi_2)$.*
They are strongly equivalent, written $\Pi_1 \sim \Pi_2$, iff for every PNE Γ, $\Gamma \cup \Pi_1 \sim_w \Gamma \cup \Pi_2$.

In [11, 12] it is proven that $\Pi_1 \sim \Pi_2$ iff $HT \vdash \Pi_1 \leftrightarrow \Pi_2$, where HT is the Here and There Logic, namely the logic obtained by adding to the intuitionistic logic Int the axiom schema $not\ A \vee not\ not\ A$.

[1] In [7], $\wedge$ is denoted by "," and $\vee$ by ";".

3 The Logic of Information Terms

In this section we briefly present the logic of information terms, a variant of the logic F_{cl} introduced in [8], with the purpose of discussing the relationship between classical and constructive logic. F_{cl} is an intermediate constructive logic with a constructable negation $\neg$ (related to Nelson's negation [9]) and an operator T, used to represent classical truth: $T(F)$ is provable in F_{cl} iff F is provable in classical logic. We have used a variant of F_{cl} to define an OO modeling language [10], where the information content of a system is modeled by *information terms*. In this paper we present another variant, based on information terms, which is indicated by $Fn^{\rightarrow,\neg}$. Our purpose is to study the relationship with answer set semantics. In $Fn^{\rightarrow,\neg}$ we replace the T-operator of F_{cl} by the double negation *not not*. That is, instead of considering strong negation $\neg$ and the T-operator, we have two negations: $\neg$ and *not*. [2] For conciseness, we consider the fragment $Fn^{\rightarrow}$ without strong negation $\neg$. In this section we present the constructive semantics of $Fn^{\rightarrow}$, which is monotonic. Default reasoning (giving rise to non-monotonicity) will be considered in the next section.

First we consider the fragment Fn of $Fn^{\rightarrow}$ without implication. The syntax of Fn is the same of nested expressions. The semantics of Fn is built on top of classical logic. A *(classical) interpretation* X is a set of atoms and $X \models F$ is defined as in classical logic (see Fig. 1). The formula $a \vee b$ could be considered as the query "which between a and b holds in $a \vee b$?". If the answer is a, we write $[1, \mathrm{T}] : a \vee b$ ("the subformula 1 is true"), while $[2, \mathrm{T}] : a \vee b$ means that the answer is b. The lists $[1, \mathrm{T}]$ and $[2, \mathrm{T}]$ are the possible information terms for $a \vee b$. We write $\tau : F$ to indicate that τ is an information term of type F and we call the pair $\tau : F$ *piece of information*. We associate with a piece of information $\tau : F$ a set $ans(\tau : F)$, indicating the answers that one can obtain from it. The typing relation ":" and the related *ans*-sets are defined in Fig. 2, where S stands for a *simple formula*, namely an atom, $\bot$, $\top$ or $not\, F$.

$$
\begin{array}{rcl}
\mathrm{T} : S \text{ with } S \text{ a simple formula} & & ans(\mathrm{T} : S) = \{S\} \\
[\tau_1, \tau_2] : F_1 \wedge F_2 \quad \text{iff} \quad \tau_1 : F_1 \text{ and } \tau_2 : F_2 & & ans([\tau_1, \tau_2] : F_1 \wedge F_2) = \bigcup_{k=1}^{2} ans(\tau_k : F_k) \\
[i, \tau_i] : F_1 \vee F_2 \quad \text{iff} \quad i \in \{1, 2\} \text{ and } \tau_i : F_i & & ans([i, \tau_i] : F_1 \vee F_2) = ans(\tau_i : F_i)
\end{array}
$$

Figure 2. Information terms for Fn and their answers

A formula $not\, F$ has a unique piece of information $\mathrm{T} : not\, F$. This means that we do not consider it as a query, but as a constraint: we only say that F cannot be true. For example, $not\, not(a \vee b)$ has a unique piece of information $\mathrm{T} : not\, not(a \vee b)$ with answer $\{not\, not(a \vee b)\}$, meaning that a and b cannot be both false. In contrast, $a \vee b$ has pieces of information $[1, \mathrm{T}] : a \vee b$ with answer $\{a\}$ and $[2, \mathrm{T}] : a \vee b$ with answer $\{b\}$. The set $ans(\tau : F)$ may contain $\bot$, $\top$, atoms and negated formulas, namely, formulas of the kind $not\, A$.

One can prove:

[2] In F_{cl} one can define $not\, A$ as $T(\neg A)$.

Theorem 1. *Let F be a nested expression and X be an interpretation. Then, $X \models F$ iff there is a piece of information $\tau : F$ such that $X \models ans(\tau : F)$.*

Now we introduce $Fn^{\rightarrow}$. Constructive implication is defined according to the BHK interpretation [13] of $\rightarrow$, where a proof of $A \rightarrow B$ is seen as a map from proofs of A into proofs of B. Here, instead of proofs, we have information terms. Let us indicate by $\textsc{it}(F)$ (the information type of F) the set $\{\tau \mid \tau : F\}$. The information terms of $Fn^{\rightarrow}$ are obtained by enriching the language of Fn with $\rightarrow$ and defining $\tau : F_1 \rightarrow F_2$ as in Fig. 3.

$$[\,[\tau_1^1, \tau_1^2], \ldots, [\tau_n^1, \tau_n^2]\,] : F_1 \rightarrow F_2 \text{ iff}$$
$$\{\tau_1^1, \ldots, \tau_n^1\} = \textsc{it}(F_1) \text{ and } \{\tau_1^2, \ldots, \tau_n^2\} \subseteq \textsc{it}(F_2)$$

Figure 3. Information terms for $F_1 \rightarrow F_2$

That is, the terms of $\textsc{it}(F_1 \rightarrow F_2)$ are lists representing the function $f : \textsc{it}(F_1) \rightarrow \textsc{it}(F_2)$. In the semantics we represent such functions by lists, to abstract from their implementation. We use the more intuitive notation $f = [\tau_1^1 \mapsto \tau_1^2, \ldots, \tau_n^1 \mapsto \tau_n^2]$, and we write $f(\tau) = \tau'$ to indicate that $\tau \mapsto \tau' \in f$. Let X be an interpretation. In Fig. 4 we define the realizability relation $X \,|\!\models \tau : F$. If $X \,|\!\models \tau : F$, we say that X is a *model* of the piece of information $\tau : F$.

1. $X \,\|\!\models \textsc{t} : S$	iff	$X \models S$, with S a simple formula
2. $X \,\|\!\models [\tau_1, \tau_2] : F_1 \wedge F_2$	iff	$X \,\|\!\models \tau_1 : F_1$ and $X \,\|\!\models \tau_2 : F_2$
3. $X \,\|\!\models [i, \tau_i] : F_1 \vee F_2$	iff	$X \,\|\!\models \tau_i : F_i$
4. $X \,\|\!\models \tau : F_1 \rightarrow F_2$	iff	for $\tau' \mapsto \tau'' \in \tau$, $X \,\|\!\models \tau' : F_1 \Rightarrow X \,\|\!\models \tau'' : F_2$

Figure 4. The realizability relation for $Fn^{\rightarrow}$

We conclude the section with the main results about $Fn^{\rightarrow}$, which can be proven as the corresponding theorems for F_{cl}.

Theorem 2. *Let F be a formula of $Fn^{\rightarrow}$ and X an interpretation. Then $X \models F$ iff there is a piece of information $\tau : F$ such that $X \,|\!\models \tau : F$.*

As a consequence, we get:

Corollary 1. *Let F be a formula of $Fn^{\rightarrow}$. Then F is classically valid iff, for every interpretation X, there is $\tau_X : F$ such that $X \,|\!\models \tau_X : F$.*

The above theorem links the realizability semantics and the semantics of classical logic. The following definition introduces the constructive semantics of $Fn^{\rightarrow}$.

Definition 3 (Constructive validity). *A formula F of $Fn^{\rightarrow}$ is constructively valid iff there is a piece of information $\tau : F$ such that, for every interpretation X, $X \models \tau : F$.*

The difference with respect to classical validity is that now the piece of information $\tau : F$ depends only on F and not on X. As an example, $a \vee not\, a$ is not constructively valid. Indeed, there is no piece of information that can be realized in all interpretations: $[1, \mathrm{T}] : a \vee not\, a$ is false in $\emptyset$, while $[2, \mathrm{T}] : a \vee not\, a$ is false in $\{a\}$. In contrast, if F is a classically valid formula, the piece of information $\mathrm{T} : not\, not\ F$ is true in every interpretation, i.e., classical validity can be expressed by double negation[3]. To show the constructive validity of an implication $F_1 \to F_2$ we have to exhibit a "realizability preserving function" (*rp-function*), namely a function $f : \mathrm{IT}(F_1) \to \mathrm{IT}(F_2)$ that satisfies point 4 of the definition in Fig. 4. One can show that every intuitionistically provable formula $F_1 \to F_2$ has an rp-function, i.e., is constructively valid in $Fn^{\to}$. As an example, $(A \vee B) \wedge (A \vee C) \to A \vee (B \wedge C)$ has the rp-function defined by the equations:

$$\begin{aligned} f([\,[1, \tau_A], \tau_2\,]) &= [1, \tau_A] \\ f([\,\tau_1, [1, \tau_A]\,]) &= [1, \tau_A] \\ f([\,[2, \tau_B], [2, \tau_C]\,]) &= [2, [\tau_B, \tau_C]\,] \end{aligned}$$

where $\tau_A : A$, $\tau_B : B$, $\tau_C : C$, $\tau_1 : (A \vee B)$, $\tau_2 : (A \vee C)$. Furthermore, the following non-intuitionistic implications are constructively valid in $Fn^{\to}$ (where S *must be a simple formula*):

$$\begin{array}{ll} \text{KP:} & (not\, F \to A_1 \vee A_2) \ \to\ (not\, F \to A_1) \vee (not\, F \to A_2) \\ \text{nnS:} & not\, not\, S \ \to\ S \end{array}$$

The axiom KP is the Kreisel and Putnam principle [2,4]. The axiom nnS gives rise to a non-standard logic, i.e., a logic where the uniform substitution principle does not hold. More precisely, we only admit the instances of nnS obtained by replacing p with a simple formula[4]. The rp-function f_{KP} for KP maps $[[\mathrm{T}, [i, \tau_i]]] : not\, F \to A_1 \vee A_2$ into $[i, [[\mathrm{T}, \tau_i]]]$ (where $i \in \{1, 2\}$ and $\tau_i \in \mathrm{IT}(A_i)$). It can be represented by the equation

$$f([\,\mathrm{T} \mapsto [i, \tau_i]\,]) \ =\ [\,i, [\mathrm{T} \mapsto \tau_i]\,]$$

The rp-function for nnS is the identity map $[\mathrm{T} \mapsto \mathrm{T}]$. Now, let $\mathscr{C}_{Fn^{\to}}$ be the calculus of intuitionistic logic enriched with the above axioms KP and nnS. In [8] it is proved that:

Theorem 3 (Validity and Completeness). *A formula F is constructively valid iff it can be proven in $\mathscr{C}_{Fn^{\to}}$.*

Furthermore, one can give calculi that support the *proofs as programs* paradigm: a proof with assumptions Γ and consequence F represents a program (in a suitable operational semantics) computing an rp-function mapping pieces of information of Γ into pieces of information of F.

We conclude this section by setting up the problem of *uniformly answerable queries* and by showing that they correspond to the constructive consequences of a theory T. An information term for a theory $T = \{F_1, \ldots F_n\}$ is a n-pla $\overline{\tau} = \langle \tau_1, \ldots, \tau_n \rangle$, where, for $1 \leq i \leq n$, $\tau_i : F_i$. We say that Q is uniformly answerable in T iff there is an effective way of extracting from every piece of information $\overline{\tau} : T$ a piece of information $\tau' : Q$, in such a way that the the interpretations satisfying $\overline{\tau} : T$ satisfy also $\tau' : Q$. By the proofs-as-programs property of the calculus, a proof of $T \vdash Q$ in $\mathscr{C}_{Fn^{\to}}$ is such an algorithm. By the completeness results, Q is uniformly answerable iff it is provable in $\mathscr{C}_{Fn^{\to}}$.

[3] As in intuitionistic propositional logic.

[4] If we take any instance of nnS, we collapse into classical logic.

4 Answer Sets for Formulas and for Pieces of Information

Here we discuss a notion of answer set of a piece of information and we compare it with answer sets as defined in Section 2. We consider nested expression; the extension to programs is part of our future work and will be briefly addressed in the conclusions. Let X be an interpretation (namely, a set of atoms) and $\mathscr{F}$ be a set of nested expressions. We write $X \models \mathscr{F}$ to say that, for every $F \in \mathscr{F}$, $X \models F$ (in particular, $X \models \emptyset$); $\mathscr{F}^X$ denotes the set of F^X such that $F \in \mathscr{F}$. We remark that $\tau : F$ iff $\tau : F^X$. Firstly, we prove some properties of answer sets and reduct operation.

Lemma 1. *Let F be a nested expression and X be an interpretation.*

(i). $X|\!\models \tau : F$ *iff* $X \models ans(\tau : F)$.
(ii). $X|\!\models \tau : F$ *iff* $X|\!\models \tau : F^X$.
(iii). $ans(\tau : F)^X = ans(\tau : F^X)$.

Proof. Point (i) and (ii) can be easily proved by induction on F.

We prove (iii) by induction on F. If F is an atom, (iii) is immediate.
Let $F = A \wedge B$. Then:

$$\begin{aligned} ans((\tau_A, \tau_B) : A \wedge B)^X &= ans(\tau_A : A)^X \cup ans(\tau_B : B)^X \\ &= ans(\tau_A : A^X) \cup ans(\tau_B : B^X) \\ &= ans((\tau_A, \tau_B) : A^X \wedge B^X) \\ &= ans(\,(\tau_A, \tau_B) : (A \wedge B)^X\,) \end{aligned}$$

where the second equivalence follows from the induction hypothesis. The case $F = A \vee B$ is similar.
Let $F = not\ A$ and assume $X \models A$. Since $(not\ A)^X = \bot$ and $\bot^X = \bot$, we get:

$$ans(\mathrm{T} : not\ A)^X = \{\, not\ A \,\}^X = \{\bot\} = ans(\mathrm{T} : \bot) = ans(\mathrm{T} : (not\ A)^X)$$

The case $X \not\models A$ (which yields $(not\ A)^X = \top$) is similar. □

Now, we study the relationship between answer sets and pieces of information.

Theorem 4. *If X is an answer set of F, then there is a piece of information $\tau : F$ such that X is a minimal model of $\tau : F$.*

Proof. Since $X \models F^X$, by Theorem 2 there is a piece of information $\tau : F^X$ such that $X|\!\models \tau : F^X$, which implies, by Lemma 1(ii), $X|\!\models \tau : F$. To prove the minimality of X, let $X' \subseteq X$ such that $X'|\!\models \tau : F$; we show that $X' = X$. By Lemma 1(i), we have both $X \models ans(\tau : F)$ and $X' \models ans(\tau : F)$. We show that $X' \models ans(\tau : F)^X$. Let $H \in ans(\tau : F)$. Clearly, $H \neq \bot$. If H is an atom or $H = \top$, then $H^X = H$, hence $X' \models H^X$. If $H = not\ A$, by the fact that $X \models H$ we have $H^X = \top$, hence $X' \models H^X$. This proves that $X' \models ans(\tau : F)^X$. By Lemma 1, we get $X' \models ans(\tau : F^X)$, hence $X' \models F^X$. Since X is a minimal model of F^X, we conclude $X' = X$.

□

To obtain the other direction of the above theorem, we need to introduce *minimality* requirements and *default reasoning* for negation. The need for the former is illustrated by the following example.

Example 1. The formula $F = (a \vee b) \wedge (a \vee c)$ has the following information terms and answer sets:

$\tau : (a \vee b) \wedge (a \vee c)$	$ans(\tau : F)$
$[\,[1, \mathrm{T}],\, [1, \mathrm{T}]\,]$	$\{a\}$
$[\,[1, \mathrm{T}],\, [2, \mathrm{T}]\,]$	$\{a, c\}$
$[\,[2, \mathrm{T}],\, [1, \mathrm{T}]\,]$	$\{a, b\}$
$[\,[2, \mathrm{T}],\, [2, \mathrm{T}]\,]$	$\{b, c\}$

The answer sets of F are $\{a\}$ and $\{b, c\}$, corresponding to "minimal" $ans(\tau : F)$.

For a positive nested expression F, we define *minimal pieces of information* as follows.

Definition 4. *Let F be a positive nested expression. A piece of information $\tau : F$ is minimal iff there is no piece of information $\tau' : F$ such that $ans(\tau' : F) \subset ans(\tau : F)$.*

In the above example, the minimal pieces of information are $[[1, \mathrm{T}], [1, \mathrm{T}]] : F$ and $[[2, \mathrm{T}], [2, \mathrm{T}]] : F$. Intuitively, the answers of the minimal pieces of information $\tau : F$ characterize those sets of atoms whose truth is strictly necessary to get evidence for F. For the other pieces of information, we do not have reasons to believe them. For a positive nested expression we can easily prove:

Theorem 5. *Let F be a positive nested expression and $\tau : F$ be a minimal piece of information for F. Then, $ans(\tau : F)$ is an answer set of F.*

W.r.t. negation, we have to introduce default reasoning for negated formulas. In Fn, *not not* p is equivalent to p, for p atomic. In contrast, this equivalence does not hold in the answer set semantics: *not not* p has no answer set, while the answer set of p is $\{p\}$. Hence, we have to adapt the semantics of Fn. We interpret negated formulas as constraints, where a constraint does not require an answer, but it simply cuts the undesired ones. That is, $\mathrm{T} : \mathit{not}\ F$ is not seen as an information, but has only the role of discarding all the answers that make F true. Adopting this view, let us define:

$$
\begin{aligned}
ans^{+}(\tau : F) &= \{\ H \in ans(\tau : F) \mid H \text{ is an atom } \} \\
ans^{-}(\tau : F) &= \{\ H \in ans(\tau : F) \mid H = \mathit{not}\, A \text{ or } H = \bot \ \}
\end{aligned}
$$

We interpret $ans^{+}(\tau : F)$ as an answer set and $ans^{-}(\tau : F)$ as the constraints that $ans^{+}(\tau : F)$ is required to satisfy.

Definition 5. *Let $\tau : F$ be a piece of information for a nested expression F. Then $ans^{+}(\tau : F)$ is the (unique) answer set of $\tau : F$ iff $ans^{+}(\tau : F) \models ans^{-}(\tau : F)$.*

Each piece of information $\tau : F$ either has no answer set or a unique answer set. Moreover, if $\tau : F$ has the answer set, then $ans(\tau : F)$ is consistent; indeed, $ans^{+}(\tau : F) \models ans(\tau : F)$, which also implies, by Lemma 1, $ans^{+}(\tau : F) \models \tau : F$. We can characterize the answer sets of the formula F as the answer sets of its "minimal" pieces of information, in the sense of the next definition.

Definition 6 (minimal). *A piece of information $\tau : F$ is minimal iff for every piece of information $\tau' : F$, $ans^{+}(\tau' : F) \subset ans^{+}(\tau : F)$ implies $ans^{+}(\tau : F) \not\models ans^{-}(\tau' : F)$.*

Intuitively, if $ans^+(\tau : F) \models ans^-(\tau' : F)$ for a a τ' such that $ans^+(\tau' : F) \subset ans^+(\tau : F)$, then $\tau : F$ gives evidence for a smaller ans^+, i.e., it is not a minimal explanation consistent with the constraints. We remark that, for positive F, Def. 6 coincides with Def. 4.

The role of minimal pieces of information is stated by the following theorems.

Theorem 6. *If $ans^+(\tau : F)$ is the answer set of a minimal piece of information $\tau : F$, then $ans^+(\tau : F)$ is an answer set of F.*

Theorem 7. *If X is an answer set of F, then there is a minimal piece of information $\tau : F$ such that $X = ans^+(\tau : F)$ and X is the answer set of $\tau : F$.*

The proofs are given at the end of this section. We firstly show some examples.

Example 2. The following examples illustrate the minimality condition and the role of ans^+ and ans^-.

- The formula $F_1 = not\ not\ p$ has no answer set. Indeed, F_1 only admits the piece of information $\mathrm{T} : F_1$. Moreover:

$$ans^+(\mathrm{T} : not\ not\ p) = \emptyset, \qquad ans^-(\mathrm{T} : not\ not\ p) = \{not\ not\ p\}$$

  and $\emptyset \not\models not\ not\ p$, hence $ans^+(\mathrm{T} : F_1)$ is not the answer set of $\mathrm{T} : F_1$.
- The formula $F_2 = p \vee not\ p$ has the following information terms τ and sets $ans^+(\tau : F_2)$, $ans^-(\tau : F_2)$:

$\tau : p \vee not\ p$	$ans^+(\tau : F_2)$	$ans^-(\tau : F_2)$
$[1, \mathrm{T}]$	$\{p\}$	$\emptyset$
$[2, \mathrm{T}]$	$\emptyset$	$\{not\ p\}$

  Both $[1, \mathrm{T}] : F_2$ and $[2, \mathrm{T}] : F_2$ are minimal pieces of information (indeed, $\{p\} \not\models not\ p$). Furthermore, $\{p\} \models \emptyset$ and $\emptyset \models not\ p$, hence $\{p\}$ is the answer set of $[1, \mathrm{T}] : F_2$ and $\emptyset$ is the answer set of $[2, \mathrm{T}] : F_2$. By Theorems 6 and 7, $\{p\}$ and $\emptyset$ are the answer sets of F_2.
- Let us consider the formula $F_3 = (p \vee q) \wedge not(p \wedge not\ q)$. We have:

$\tau : (p \vee q) \wedge not(p \wedge not\ q)$	$ans^+(\tau : F_3)$	$ans^-(\tau : F_3)$
$[\,[1, \mathrm{T}],\ \mathrm{T}\,]$	$\{p\}$	$\{\ not(p \wedge not\ q)\ \}$
$[\,[2, \mathrm{T}],\ \mathrm{T}\,]$	$\{q\}$	$\{\ not(p \wedge not\ q)\ \}$

  Both $[[1, \mathrm{T}], \mathrm{T}] : F_3$ and $[[2, \mathrm{T}], \mathrm{T}] : F_3$ are minimal. Since $\{p\} \not\models not(p \wedge not\ q)$, $\{p\}$ is not the answer set of $[[1, \mathrm{T}], \mathrm{T}] : F_3$. Since $\{q\} \models not(p \wedge not\ q)$, $\{q\}$ is the answer set of $[[2, \mathrm{T}], \mathrm{T}] : F_3$. Thus, the answer set of F_3 is $\{q\}$.
- For the formula $F_4 = (p \wedge q) \vee (p \wedge not\ q)$ we have:

$\tau : (p \wedge q) \vee (p \wedge not\ q)$	$ans^+(\tau : F_4)$	$ans^-(\tau : F_4)$
$[\,1, [\mathrm{T}, \mathrm{T}]\,]$	$\{p, q\}$	$\emptyset$
$[\,2, [\mathrm{T}, \mathrm{T}]\,]$	$\{p\}$	$\{not\ q\}$

  Both $[1, [\mathrm{T}, \mathrm{T}]] : F_4$ and $[2, [\mathrm{T}, \mathrm{T}]] : F_4$ are minimal, since $\{p, q\} \not\models not\ q$. Since $\{p, q\} \models \emptyset$ and $\{p\} \models not\ q$, the answer sets are $\{p, q\}$ and $\{p\}$.

– The formula considered in Example 1 is another illustration of the role of minimality.

By $\bigwedge ans(\tau : F)$ we denote the nested expression $\bigwedge_{H \in ans(\tau:F)} H$. For a nested formula F the *disjunctive normal form* D_F of F is the nested expression defined as follows:

$$D_F \;=\; \bigvee_{\tau \,\in\, \mathrm{IT}(F)} \big(\bigwedge ans(\tau : F) \big)$$

For instance, for the formula $F_3 = (p \vee q) \wedge not(p \wedge not\, q)$ of the previous example, we have:

$$D_{F_3} \;=\; \big(p \wedge not(p \wedge not\, q)\big) \;\vee\; \big(q \wedge not(p \wedge not\, q)\big)$$

We write $H_1 \equiv_{Int} H_2$ to mean that H_1 is equivalent to H_2 in the intuitionistic logic Int. Note that, since Int is contained in HT, H_1 and H_2 have the same answer sets.

Lemma 2. *Let F be a nested expression and X be an interpretation. Then:*

(i). $F \;\equiv_{Int}\; D_F$.
(ii). $D_{(F^X)} = (D_F)^X$.

Proof. We prove (i) by induction on F. The cases where F is an atom or a negated formula are trivial, since D_F coincides with F.
Let $F = A \wedge B$. By the induction hypothesis we have:

$$A \;\equiv_{Int} \bigvee_{\tau_A \,\in\, \mathrm{IT}(A)} \big(\bigwedge ans(\tau_A : A) \big) \qquad\qquad B \;\equiv_{Int} \bigvee_{\tau_B \,\in\, \mathrm{IT}(B)} \big(\bigwedge ans(\tau_B : B) \big)$$

By applying the de Morgan's equivalences (which transform a formula into an intuitionistically equivalent one), we get

$$A \wedge B \;\equiv_{Int} \bigvee_{\tau_A \,\in\, \mathrm{IT}(A), \tau_B \,\in\, \mathrm{IT}(B)} \big(\bigwedge ans(\tau_A : A) \;\wedge\; \bigwedge ans(\tau_B : B) \big)$$

namely

$$A \wedge B \;\equiv_{Int} \bigvee_{(\tau_A, \tau_B) \,\in\, \mathrm{IT}(A \wedge B)} \big(\bigwedge ans(\,(\tau_A, \tau_B) : A \wedge B) \big)$$

where the right-hand formula coincides with $D_{A \wedge B}$.
Let $F = A \vee B$. By the induction hypothesis, we get:

$$A \vee B \;\equiv_{Int} \bigvee_{\tau_A \,\in\, \mathrm{IT}(A)} \big(\bigwedge ans(\tau_A : A) \big) \;\vee\; \bigvee_{\tau_B \,\in\, \mathrm{IT}(B)} \big(\bigwedge ans(\tau_B : B) \big)$$

Since the right-hand formula can be rewritten as

$$\bigvee_{(1, \tau_A) \,\in\, \mathrm{IT}(F)} \big(\bigwedge ans((1, \tau_A) : F) \big) \;\vee\; \bigvee_{(2, \tau_B) \,\in\, \mathrm{IT}(F)} \big(\bigwedge ans((2, \tau_B) : F) \big)$$

we conclude $F \equiv_{Int} D_F$.

Point (ii) follows from the definition of D_F and from Lemma 1(iii), indeed:

$$D_{(F^X)} \;=\; \bigvee_{\tau \,\in\, \mathrm{IT}(F^X)} \big(\bigwedge ans(\tau : F^X) \big) \;=\; \bigvee_{\tau \,\in\, \mathrm{IT}(F)} \big(\bigwedge ans(\tau : F)^X \big)$$

and the last formula coincides with $(D_F)^X$. □

Now we can prove Theorems 6 and 7.

Proof of Theorem 6. Let us assume that $X = ans^+(\tau : F)$ is the answer set of $\tau : F$, namely $X \models ans^-(\tau : F)$; we prove that X is an answer set of F. Clearly, $X \models ans(\tau : F)$ hence, by Lemma 1, we get $X \models F^X$. Now we show that X is a minimal model of F^X. Let us assume by absurd that there exists $X' \subset X$ such that $X' \models F^X$. By Lemma 2, $F^X \equiv_{Int} D_{(F^X)} = (D_F)^X$, hence $X' \models (D_F)^X$. By definition of $(D_F)^X$, there exists $\tau' : F$ such that $X' \models ans(\tau' : F)^X$, namely:

(a). $X' \models ans^+(\tau' : F)^X$;
(b). $X' \models ans^-(\tau' : F)^X$.

Since $ans^+(\tau' : F)^X = ans^+(\tau' : F)$, (a) implies that $ans^+(\tau' : F) \subseteq X'$, hence $ans^+(\tau' : F) \subset X$. By (b), $ans^-(\tau' : F)^X$ only contains $\top$, and this means that $X \models ans^-(\tau' : F)$. This contradicts the hypothesis that $\tau : F$ is a minimal piece of information. □

Proof of Theorem 7. Let X be an answer set of F. Since, by Lemma 2, $F \equiv_{Int} D_F$, it follows that X is an answer set of D_F as well, i.e., X is a minimal model of $(D_F)^X$. By definition of $(D_F)^X$, there exists $\tau : F$ such that:

(1). $X \models ans(\tau : F)^X$;
(2). For every $X' \subset X$ and $\tau' : F$, $X' \not\models ans(\tau' : F)^X$.

Note that (1) implies that $ans^-(\tau : F)^X = \{\top\}$, hence $X \models ans^-(\tau : F)$. By (2), for every $X' \subset X$, $X' \not\models ans(\tau : F)^X$. Since $ans^-(\tau : F)^X = \{\top\}$ and $ans^+(\tau : F)^X = ans^+(\tau : F)$, it follows that for every $X' \subset X$, $X' \not\models ans^+(\tau : F)$, and this proves that $X = ans^+(\tau : F)$. Since $X \models ans^-(\tau : F)$, X is the answer set of $\tau : F$. Let $\tau' : F$ be such that $ans^+(\tau' : F) \subset X$. By (2), $ans^+(\tau' : F) \not\models ans(\tau' : F)^X$, namely $ans^+(\tau' : F) \not\models ans^-(\tau' : F)^X$. This means that $\bot \in ans^-(\tau' : F)^X$, hence $X \not\models ans^-(\tau' : F)$, thus $\tau : F$ is minimal. □

5 Conclusion

We have presented the semantics of pieces of information used in the CooML system to characterize the information content of classes, and we have compared it with the answer set semantics of the language of nested expressions. We have shown that, if in the CooML semantics we consider negated formulas as constraints, we can link pieces of information and answer sets. Introducing a suitable notion of minimal pieces of information, we get answer sets for nested expressions, as defined in [7]. To prove this fact, we have used both the results about the logic F_{cl} [8] and about strong equivalence in answer set semantics [1, 6].

The next step is to introduce programs with nested expressions. This requires to extend our analysis to rules F :- G. A first step is to study the cases where P :- Q is strongly minimally equivalent to $P \wedge Q \vee not\, Q$, i.e., the cases where we can consider the latter formula as an abbreviation of P :- Q. In the general case, the idea is to represent F :- G by information terms $\tau : \text{IT}(G) \rightarrow \text{IT}(F)$ that satisfy suitable restrictive properties.

Finally, we want to use the idea of constructively valid implication based on rp-functions, by introducing a suitable notion of answer-set realizability. This would allow us to deal with the problem of uniform answerability, related to the existence of algorithms that transform answer sets into answer sets, in a way analogous to the one mentioned

above for $Fn^{\rightarrow}$. For example, a constructive proof of $\Gamma \vdash (H \text{:-} B) \rightarrow (H' \text{:-} B')$ would provide a rp-function mapping every piece of information for $\Gamma \cup \{H \text{:-} B\}$ into a piece of information of $\Gamma \cup \{H' \text{:-} B'\}$, while preserving the answer sets.

References

1. P. Ferraris and V. Lifschitz. Mathematical foundations of answer set programming. In *We Will Show Them! (1)*, pages 615–664, 2005.
2. D. M. Gabbay. The decidability of the Kreisel-Putnam system. *J. Symbolic Logic*, 35:431–437, 1970.
3. M. Gogolla, J. Bohling, and M. Richters. Validating UML and OCL models in USE by automatic snapshot generation. *Software and System Modeling*, 4(4):386–398, 2005.
4. G. Kreisel and H. Putnam. Eine Unableitbarkeitsbeweismethode für den intuitionistischen Aussagenkalkul. *Arch. Math. Logik Grundlagenforsch.*, 3:74–78, 1957.
5. N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello. The DLV system for knowledge representation and reasoning. *ACM Trans. Comput. Log.*, 7(3):499–562, 2006.
6. V. Lifschitz, D. Pearce, and A. Valverde. Strongly equivalent logic programs. *ACM Trans. Comput. Log.*, 2(4):526–541, 2001.
7. V. Lifschitz, L. R. Tang, and H. Turner. Nested expressions in logic programs. *Ann. Math. Artif. Intell.*, 25(3-4):369–389, 1999.
8. P. Miglioli, U. Moscato, M. Ornaghi, and G. Usberti. A constructivism based on classical truth. *Notre Dame Journal of Formal Logic*, 30(1):67–90, 1989.
9. D. Nelson. Constructible falsity. *J. Symbolic Logic*, 14:16–26, 1949.
10. M. Ornaghi, M. Benini, M. Ferrari, C. Fiorentini, and A. Momigliano. A constructive object oriented modeling language for information systems. *ENTCS*, 153(1):67–90, 2006.
11. D. Pearce. A new logical characterisation of stable models and answer sets. In *Non-monotonic extensions of logic programming (Bad Honnef, 1996)*, volume 1216 of *Lecture Notes in Comput. Sci.*, pages 57–70. Springer, Berlin, 1997.
12. D. Pearce. From here to there: stable negation in logic programming. In *What is negation?*, volume 13 of *Appl. Log. Ser.*, pages 161–181. Kluwer Acad. Publ., Dordrecht, 1999.
13. A. S. Troelstra. From constructivism to computer science. *TCS*, 211(1-2):233–252, 1999.